Toward automatic GIS-video initial registration

Gaël Sourimant, Thomas Colleu, Vincent Jantet, Luce Morin, Kadi Bouatouch

Ann. Telecommun.
DOI 10.1007/s12243-011-0249-8

Toward automatic GIS–video initial registration

And application to buildings textures computation

Gaël Sourimant · Thomas Colleu · Vincent Jantet · Luce Morin · Kadi Bouatouch

Received: 12 November 2009 / Accepted: 2 March 2011

Abstract In the context of registration between videos and geographic information system (GIS)-based 3D building models—for instance in augmented reality applications—we propose a solution to one of the most critical problems, namely the registration initialization. Successful automatic 2D/3D matching is achieved by combining two context-dependent improvements. On one hand, we associate semantic information to the low-level primitives we used to reduce the problem complexity. On the other hand, we avoid false initial registration solutions by analyzing the convergence of the iterative pose computation in a RANSAC framework. We require that videos are acquired together with global positioning system measures. We also present how such a registration can be exploited, once it has been performed for the whole video. Textures of visible buildings are extracted from the images. A new algorithm for façade texture fusion based on statistical analysis of the texels color is presented. It allows us to remove from the final textures all occluding objects in front of the viewed building façades.

G. Sourimant (✉) · T. Colleu · V. Jantet
INRIA Rennes Bretagne Atlantique,
Campus Universitaire de Beaulieu,
35042, Rennes Cedex, France
e-mail: gael.sourimant@irisa.fr

L. Morin
IETR/INSA Rennes,
20 Avenue des Buttes de Coësmes,
CS 14315, 35043, Rennes, France

K. Bouatouch
Université de Rennes 1,
Campus Universitaire de Beaulieu,
35042, Rennes Cedex, France

Keywords Data registration · Urban modeling · Texture synthesis · GPS · GIS · Ransac

1 Introduction

Computation of realistic 3D models of urban environments has numerous applications in both virtual and augmented reality. A first example is the tremendous demand for high-quality 3D building models access in virtual reality applications, such as Google Earth or Microsoft's Virtual Earth. Unfortunately, in these kind of virtual tours, the photo-realistic aspect of the buildings is not always obvious. Some are textured automatically using aerial photographs, leading mainly in poor resolution or misaligned images, while others (the most famous ones) are modeled manually. Another application is the use of 3D building models within forthcoming location-based systems, such as global positioning system (GPS) navigation, where one would like to be able to add additional virtual information online and not only to offline constructed maps.

In both these complementary cases, the key point is the fusion—or registration—between real images and synthetic 3D models. It is the mandatory step if one wants either to enhance existing rough 3D models with ground-based images or to provide additional virtual information to urban videos.

1.1 Motivations

This paper explores the two sides of the registration process between videos and synthetic building models. The input data consist in a synthetic model of geo-referenced buildings (extracted from a *geographic*

Published online: 30 March 2011

information system (GIS)), storing for each building its footprint and elevation, together with videos acquired at ground level within urban areas. These videos are associated with GPS measures providing the approximate position of the camera in the geo-referenced coordinate frame.

The key problem we wish to solve can be formulated in this single question: How can we accurately register 3D models to videos in urban areas, while having poor precision on each input data type? As a matter of fact, real datasets used in most envisioned applications are not necessarily precisely modeled, calibrated, or localized. For that reason, we wish to keep from end-to-end unconstrained datasets (Section 2), contrary to the majority of existing approaches on this subject. The uncertainties are here of different natures. The GIS building models have been estimated using aerial photographs. Their façades are thus approximated using simple planes, leaving the micro-structures (windows, doors, etc.) unmodeled. Second, GPS measures in urban areas usually suffer from poor accuracy, mainly due to multiple signal reflections on walls and narrow satellite coverage. Last, if we imagine a collaborative application where hundreds of users could collaborate to acquire image data and enhance building models, we cannot constrain the camera to be calibrated (in terms of its internal parameters).

1.2 Contributions

This paper presents a set of methods that allow to register 3D building models to videos and also to enhance these models by synthesizing photo-realistic textures of their façades using ground level imagery. All these methods have been designed to take into account the fact that the input datasets have poor accuracy. The major contributions of this paper are the following:

1. The first contribution of this paper is the registration initialization, that is, the computation of the camera pose (position and orientation) in a geo-referenced coordinate system for the first image of a video. Contrary to most of existing similar systems, no orientation cue is required: Only a rough estimation of the position is used as an input. Moreover, semantic information extraction is performed on the images to build correct sets of matches between images and 3D models and thus compute the initial pose of the camera in a robust way.
2. The second contribution is the exploitation of the proposed registration to enhance existing buildings 3D models. We developed a new robust hybrid texture fusion algorithm for façades textures synthesis that summarizes existing methods, which do not generally aim at solving each problem related to this concern.

Pose tracking, that is, computing the pose of the camera for all images, has already been presented in [17]. As a matter of fact, we will only remind it briefly in this paper. This pose tracking scheme is robust to partial visibility of the buildings and to occluding objects, even considering the coarse geometry of the models.

1.3 Related works

In this section, we define more precisely what 2D/3D registration is and review existing methods to perform such tasks. So as to fuse video images and GIS, both these data have to be matched. For each image of the sequence, one has to determine the position and the orientation of the camera in the geo-referenced frame, in such a way that the perspective projection of the GIS 3D models be aligned to the building contours in the images. This is the camera pose computation task that is called *registration* in this paper.

Registration initialization consists in estimating simultaneously camera pose for the first image and a set of matched 2D–3D primitives. It is a complex problem to which many contributions can be found in the literature. One solution consists in eliminating one of the two unknowns (either pose or matching) thanks to the user intervention or by using more elaborated acquisition devices. In [5, 8, 16], the 2D–3D correspondences are provided by the user. In [15, 19], it is the pose itself that is measured using more precise navigation devices (GPS + inertial sensor pack). Other solutions have been proposed based on more complex models, such as textured 3D models [15] or models acquired through a 3D scanner [10].

The 3D models we used in this work only describe the buildings outlines. In such case, two types of methods exist to jointly estimate pose and matching, assuming a rough estimate of the pose is available. The first family of estimation methods is based on the RANSAC algorithm [6], which is efficient if both the number of primitives and the number of potential outliers (i.e., false matches) are small. The second method is based on the minimization of an energy functional [3] which is not assured to converge due to the non-linearity of the cost function. Both these methods do not fit naturally in our context since we have generally a large amount of outlier matches, which can also be highly corrupted by noise either in 2D or 3D space. We present adapted versions of these methods to our registration task in Section 3.

Basing on such a registration, the models can be enhanced using the images extracted from the video. For instance, high-quality textures of the façades can be computed, which is not possible when only aerial images are used. The intrinsic redundancy of the video can be exploited to compute the best possible textures: A single façade may be viewed in several images with different points of view, and the final texture is computed by integrating all the registered images in which the façade is visible.

Several issues generally arise from such façade textures generation algorithms. First, the 2D space in which the textures are computed has to be determined. They can be expressed either in their principal plane space [11, 20] or in an arbitrary reference image space [9, 12]. The former solution is often preferred since it allows more flexible (and yet undefined) uses of the building textures (e.g., for virtual views synthesis). Second, a building may be only partially visible in each image or masked by another building. These are modeled occlusions, meaning that they are induced by known objects: the camera field of view in the first case, other objects present in the 3D model in the second one. One can deal with these occlusions by the use of masks derived from the image/model registration [11, 13, 20]. Third, nonmodeled occlusions (i.e., occlusions generated by objects which are not present in the 3D database) are generally suppressed using either robust tools based on median luminance measures [9, 12] or iterative processes based on correlation masks [20]. Finally, some studies aim at dealing with the spatial resolution of the different textures within a single stack [11, 13, 20], filling unknown zones [9, 14], or treating illumination variations [20].

Despite the fact that most texture fusion issues are identified, none of these methods solves all the problems. This paper thus explores a new *robust hybrid texture fusion* algorithm (RHTF) that aims to solve all these major texture fusion issues.

1.4 Overview

Figure 1 outlines our video–GIS registration framework. Videos are acquired together with GPS measures, which permit to approximately determine the camera position in the geo-referenced frame, in which the 3D models are defined. We briefly expose in Section 2 the issues induced by our choice to keep unconstrained and approximate data.

By relating 2D primitives extracted from the images to 3D features belonging to the GIS models, the camera pose (i.e., position and orientation) is estimated for each image. The goal is that the perspective projection of the GIS in the camera coordinate frame is aligned to the buildings in the images. The peculiar case of initial pose computation is exposed in Section 3. How this pose is estimated in all remaining images is reminded in Section 4.

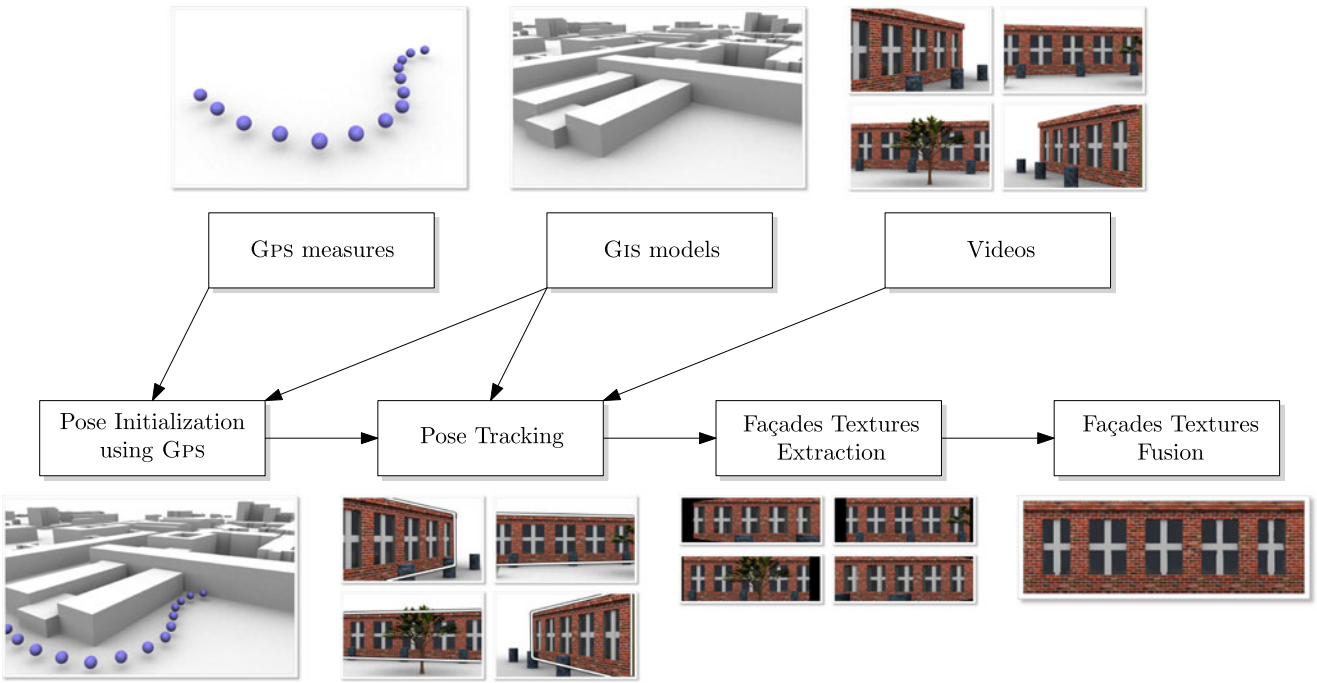


Fig. 1 Overview of our video–GIS registration framework

Finally, we demonstrate in Section 5 that such 2D/3D registration can be exploited to generate high visual quality textures of building façades using our robust hybrid texture fusion algorithm and which can be reused in convincing virtual reality application.

2 Datasets

As pointed out in Section 1, all input data are intrinsically prone to inaccuracies and errors. This is a strong constraint on this work. We identify here in what extend input data suffer from lack of precision.

2.1 GIS data

GIS data are designed to hold layers of any type of geographic information (hydrographic maps, road maps, etc.). We use a specific layer defining simple 3D models of buildings. These are specified in the UTM geo-referenced frame, using closed 2D point lists defining the buildings footprint and their altitude above sea level, together with the buildings height (both expressed in meters). This permits to generate simple polyhedral models representing large urban areas (see Fig. 2 to view a rendering of such a database). Several limitations arise from such a representation. First, buildings façades are defined as an extrusion of their footprint. As such they are perfectly planar and do not model geometric details such as doors or windows. Moreover, curved buildings are often badly modeled due to lack of points to describe the corresponding footprint. Second, holes-like topologies, such as arches, are not modeled. Last, the footprint being expressed in the horizontal UTM frame, buildings constructed on nonglobally planar ground suffer from erroneous footprints.

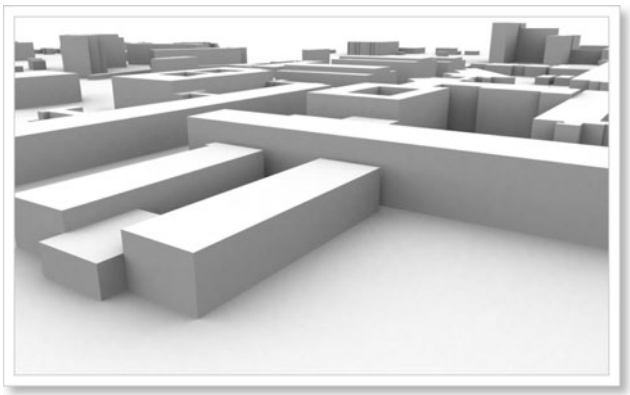


Fig. 2 Example of rendered GIS models. Notice the simple topology of such buildings representation

Table 1 Fixed point GPS measures statistics

	σ_X (m)	σ_Y (m)	σ_Z (m)	Imprecision (m^2)
Occlusions	3.15	4.29	6.90	10.61
Clear view	0.62	0.95	1.76	0.46

2.2 Videos and associated camera

Since one of the targeted applications is collaborative modeling, the registration framework should not be restricted to specific user acquisition devices. As a consequence, the less hypotheses are made on the camera, the better. The classical pinhole projective model is chosen, and the basic camera internal parameters are supposed to be available: focal of the lens (in meters), image dimensions (in pixels), and charge-coupled device captor dimensions (in meters). This information is sufficient to get an approximation of the camera focal lengths f_x and f_y. The central point (u_0, v_0) is arbitrarily set at the image center, since errors on the central point estimation have a negligible influence in our context [7].

2.3 GPS measures

GPS measures are acquired jointly to videos so as to localize images with respect to the 3D model. However, GPS does not provide and exact position location. Inaccuracies arise mainly from occluding environment in urban areas, where less satellites are visible and where the signal can be delayed with multiple reflections on buildings. To estimate this imprecision, we measured fixed GPS positions (i.e., without moving the device) either with clear view or occluding environment[1] during 30 min. These measures are summarized in Table 1. Position accuracy at ground level is measured as the ellipse area which axes are the position standard deviations σ_X and σ_Y. We observe two facts from these results. First, we can expect ground precision to go from less than a meter to a little more than 10 m. Second, vertical precision is much poorer than horizontal one. As a consequence, one should not take vertical measures into account if an alternative is available.

Because GPS measures are considered in our framework as approximate initial camera positions, we do not synchronize them with the video frames with hardware tools. The two devices are stopped at the (approximate) same time so that GPS and video timings can be roughly aligned.

[1] Both measures take advantage of the satellite-based augmentation system corrections

3 Initial pose computation

This section presents a solution to our most challenging issue, namely the initialization of the registration between the video and the GIS-based 3D models.

3.1 Exploiting keyframes

In order to initialize registration between a GIS-based 3D model and a corresponding video, the only available data are the GPS measures acquired together with the video. GPS measures provide an approximate *position* of the camera within the GIS coordinate system. However, we cannot infer any information about the camera *orientation*. In order to get a precise estimate of both position and orientation, a two-step process is performed:

1. *Approximate pose estimation.* Approximate relative motion and orientation between two given camera positions is estimated using the images alone, with a *structure from motion* (SfM) approach. The computed translation is then matched to the given translation between the two corresponding GPS measures. This matching provides a rough orientation of the camera for the first image.
2. *Pose refinement.* Approximate pose is then exploited to detect 3D lines within the model and match them to 2D lines extracted from the images. These 2D lines are constrained by the image context: They either correspond to borders between buildings and ground, buildings and sky, or to vertical features within the buildings. Camera pose is computed via a RANSAC procedure which seeks to find the best matches that minimize the Euclidean error between extracted 2D lines and projected 3D lines. The problem of buildings constructed on a nonplanar surface is partly solved in this approach with the attribution of different weights to lines, depending on their associated context.

This work has already been described with more details in [1]. The major limitation of this approach is that it requires to choose a *keyframe* within the video to compute the first approximation of the camera pose in the SfM framework. In [1], the keyframe is selected manually.

3.2 Automatic initial registration

We aim at minimizing even more user intervention in the registration process and achieve completely automatic procedure. Arbitrary choice for the keyframe cannot be a relevant choice since success of the approximate pose estimation highly depends on input data. For instance, GPS measures can be too noisy, or baseline may be insufficient to correctly compute relative pose between the first image and the keyframe.

We thus propose a *supervised* algorithm for initial pose computation, where user intervention is limited to validation or rejection of a computed pose. If the computed pose is rejected, a new one is proposed to the user, using another keyframe. This method is based on the following assumption: Given a keyframe, which criteria allow to determine whether the estimated pose is a viable one? The registration procedure becomes sequential, meaning that we test each image of the video as a keyframe, one after the other, using the algorithm described in [1]. Several criteria are used to determine whether a keyframe is a valid one for initial pose computation. Once one of these criteria is invalidated, the following image is used as a keyframe. The different criteria are now explained in the paragraphs below.

3.2.1 Epipolar geometry criterion

First of all, we evaluate the quantity of epipolar residual induced by the estimated fundamental matrix $\mathbf{F}$, computed using a robust RANSAC process. If this residual is too high, epipolar geometry is considered too poorly estimated to provide a trustworthy relative pose for the cameras. Otherwise, initial pose is inferred by pose extraction from the fundamental matrix and from the camera estimated intrinsic parameters and then by identification to the GPS motion between the two frames.

This problem is equivalent to the definition of two suitable images to compute a viable initial frame (camera poses and scene structure) in a SfM framework. To our knowledge, this is still an open problem. Here, thresholding the epipolar residual does not ensure to find two good initial poses. In fact, a small epipolar residual is a necessary but not sufficient condition to ensure a correct initial frame. Instead, it directly removes potential gross errors. So the initial pose has to be tested against other criteria, as presented in the paragraphs below.

3.2.2 Primitives extraction criterion

Once approximate pose is estimated, the keyframe is kept only if enough projected 3D lines can be extracted and potentially matched to 2D image lines for the precise pose computation process. The number of required

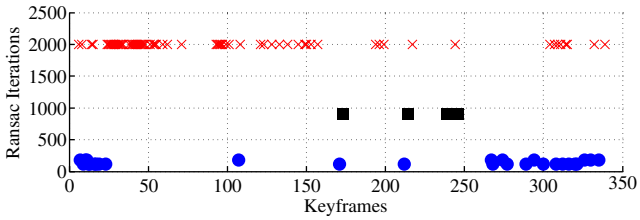


Fig. 3 RANSAC iterations vs. selected keyframe. *Blue disks* indicate a valid proposed solution, *black squares* an invalid proposed solution, and *red crosses* an absence of proposed solution due to a too large number of RANSAC iterations. Notice that some frames do not pass the epipolar or primitives test and as such are not evaluated against the RANSAC procedure

line matches and mandatory geometric configuration of these matches are described with more details in [1].

3.2.3 Robust matching criterion

Given 2D and 3D lines, a RANSAC-based algorithm iterates over the whole possible matches to remove outliers. We experimentally noticed that the success of this phase is highly correlated to the number of RANSAC iterations and to the final number of valid matches the RANSAC procedure finds. Actually, this number of iterations decreases when the probability to find a viable solution increases. Thresholding the number of RANSAC iterations allows to validate or reject the proposed keyframe.

The result of the keyframe selection is illustrated in Fig. 3. The graph illustrates the number of RANSAC iterations performed for each keyframe that passed the epipolar and primitives checks. Blue disks indicate a valid proposed solution, black squares indicate an invalid proposed solution, and red crosses indicate an absence of proposed solution due to a too large number of RANSAC iterations. Here the validity is determined by a user decision. The proposed pose corresponding to one of the black squares is illustrated in Fig. 4 (center) while a pose corresponding to one of the blue disks is depicted in Fig. 4 (right). We explain now how proper thresholding of the RANSAC iterations can be set up.

Fig. 4 Automatic initial registration: example of failed pose computation with arbitrary keyframe selection (*left*), false proposition (too many RANSAC iterations, *center*) and obtained registration validated by the user (small number of RANSAC iterations, *right*)

This relationship between correct solutions and convergence speed can be explained by the way iterations are dealt within the RANSAC algorithm. At each step, the maximum number of iterations is updated as formalized in the following equation:

$$N = \frac{\log(1-p)}{\log(1-w^n)}, \tag{1}$$

with N the maximum number of iterations, p the (fixed) desired probability to compute a correct solution, n the model dimension (i.e., the number of required matches to compute a pose in our case), and w the probability that a random 2D/3D match is a correct one. From Eq. 1 we deduce that:

$$w = \left(1 - e^{\frac{\log(1-p)}{N}}\right)^{\frac{1}{n}} \tag{2}$$

This means that a given approximate pose associated to potential 2D/3D matches will lead to a probability w to randomly pick a correct 2D/3D match. A high value of w means our problem is well posed, that a good solution can easily be computed, and thus that the RANSAC algorithm will perform with few iterations. This is illustrated in Fig. 5, where we set p to 0.99 (a standard value) and n to 4 [1]. Back to Fig. 3, the good solutions (blue dots) are proposed when the initial approximate pose induce a probability w equal to 0.4442 or 0.3998 (resp. $N = 116$ and $N = 178$), while the false proposed solutions (black dots) correspond to a probability w equal to 0.2666 ($N = 909$). So thresholding the maximum number of iterations is somewhat equivalent to invalidate the cases where the initial pose is too badly estimated to be corrected through the RANSAC procedure.

To summarize, if a keyframe successfully passes these three tests, the computed pose is submitted to the user, who can accept or reject it. In case of rejection, a new keyframe is determined and the procedure iterated. With our datasets however, the rejection case arises much rarer than the acceptance case, since most of the keyframes leading to a very incorrect approxi-

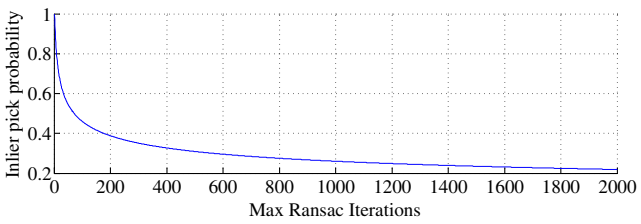


Fig. 5 Probability of picking a good 2D/3D match given an approximate initial pose vs. max. RANSAC iterations

mate pose are discarded before the RANSAC precise pose computation step.

4 Pose tracking

Once the pose of the camera is estimated for the first image of the video, we seek to compute it for all remaining images. With the assumption of small inter-image motion, this pose computation is performed by pose tracking from the previous images. We present here a small summary of our tracking method. The reader can refer to [17] for more details.

Many methods to perform such registration are already described in the literature (see for instance [4]). The method we use is a variation of the virtual visual servoing algorithm described in [2]. Pose computation by visual servoing requires matches between 3D primitives belonging to the model to be registered and projected 2D features extracted from the images. We use points as matching primitives, since they are simple to model, extract, and track. So as to track the 3D model (and thus the pose) from one image to the other, we use a point transfer scheme. Relevant points are a priori those viewed both in the model and the images, namely points belonging to the buildings façades.

Pose tracking algorithms being naturally prone to drift, they have to be integrated in a robust framework to perform correctly. To this end, ground points are introduced in the tracking process in addition to façade points. Since no 3D information about the ground is supposed to be known (we do not assume the GIS holds any digital elevation model), ground points 3D coordinates are computed using a Delaunay triangulation of the surrounding building footprints.

Moreover, the servoing command law used to compute the pose is augmented with a M-estimator as proposed by [2]. The function to be minimized then becomes:

$$\mathbf{v} = -\lambda(\mathbf{DL})^{+}\mathbf{D}(\mathscr{P}(\mathbf{X}) - \mathbf{x}), \tag{3}$$

with $\mathbf{D} = \mathrm{diag}(w_1, ..., w_N)$ being the weights computed during the M-estimation and associated to each 2D–3D point match. The cost function used here is a Cauchy robust function. This is justified by the fact that input matches can be highly noisy, so the robust cost function used should be permissive enough (as opposed to a Tuckey robust function for instance). The vector $\mathbf{v}$ models the desired camera pose, $\mathscr{P}(\mathbf{X}) - \mathbf{x}$ is the 2D Euclidean difference for a given point match between its projected 3D part and the measured 2D counterpart, and $\mathbf{L}$ describes the interaction matrix depending both on the projected primitives and the relative depth between the camera and the viewed model (here the GIS building).

Fig. 6 Tracking results for two real sequences, the *green* part being the superimposed estimated ground model

Tracking results for two sequences are illustrated in Fig. 6. The registered 3D model is depicted with a white outline, while the green part of the images corresponds to the estimation of the local ground plane. Even with occluding objects, or with some building disappearing and reappearing, pose tracking remains sufficiently correct to superimpose the building model outline to the video.

5 Building textures synthesis

Once pose tracking has been performed, 2D video data and 3D buildings data are registered. They can then be combined, for instance to enrich 3D models using textures extracted from the video frames. The problem we aim at solving here consists in computing the final texture T of each visible façade. Each façade f visible in image I_k can be associated to a corresponding texture T_k^f.

This texture is generally incomplete, partially masked by foreground objects nonmodeled in the GIS database or simply other buildings. Moreover, it can suffer from blur depending on the texture extraction method and the camera geometric configuration regarding the 3D models. Computation of T is performed in two steps: extraction of textures T_k^f and construction

of the corresponding textures stack, then computation of T by fusing the textures stack in a texel-wise fashion. We differentiate in this text the *pixels*, which are the base units of the images, and the *texels*, which are the base units of the textures. The two steps are described more precisely in the sections below.

5.1 Extraction from images

Let I_k ($k \in \{1..n\}$) be an image from the original video. We suppose that m façades are visible in I_k. Using this image, the camera pose and the 3D model, the goal is to compute the m textures T_k^f corresponding to fronto-parallel images of these façades. The dimension ratio of T_k^f is the same as the façade dimensions ratio in the 3D model. A scale factor η (chosen by the user) links the metric domain to the texel domain.

Thanks to image-model registration, the coordinates of the four corners of each façade can be computed in any image frame. These coordinates are denoted $\mathbf{x} = [u_i\ v_i]^\top$, $i \in \{1..4\}$. To compute the (homographic) transformation that allows to transfer in the texture space of T_k^f, these points are matched with the four corners of T_k^f, which coordinates are $\mathbf{x}' = [0/u'_j\ 0/v'_j]^\top$, $j \in \{1..4\}$. If w (resp. h) is the width (resp. the height) of the façade f, we then have $u'_j = \eta w$ and $v'_j = \eta h$. The matching process is illustrated in the Fig. 7.

The transfer from $\mathbf{x}$ to $\mathbf{x}'$ is performed using a 3×3 homography matrix ($\mathbf{x}' \sim \mathbf{H}_k^f \mathbf{x}$), which is defined by solving the linear system formed by these 2D matches.

Once homographies $\mathbf{H}_k^f$ are estimated, textures T_k^f are computed using the inverse transform $(\mathbf{H}_k^f)^{-1}$. For each texel $[u'\ v']^\top$ of T_k^f, its corresponding position $[u\ v]^\top$ in I_k is defined as $[u\ v\ 1]^\top \sim (\mathbf{H}_k^f)^{-1}[u'\ v'\ 1]^\top$.

Since in general $[u\ v]^\top$ is not an integer pixel position, the color of $[u'\ v']^\top$ is estimated by interpolating the colors of I_k's pixels in the neighborhood of $[u\ v]^\top$. We suggest to use a bi-cubic interpolation, since it better preserves the image contours than the bi-linear interpolation.

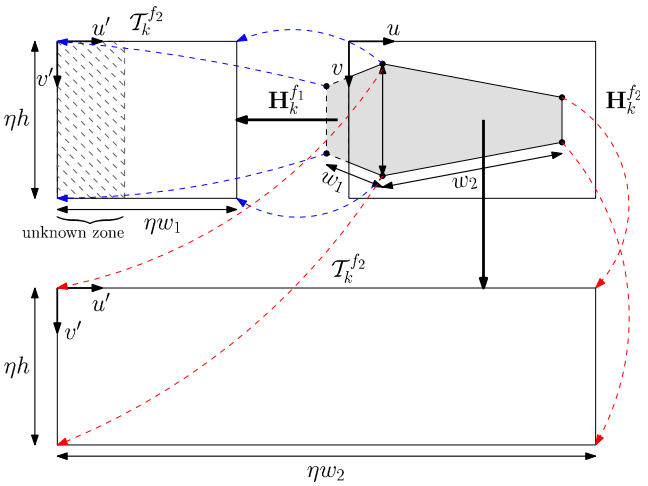


Fig. 7 Matching the image and texture coordinates to compute the textures

5.2 Texture fusion

For a given façade f, k textures T_k^f are computed, k being the number of images in which f is visible at least partially. They are stored into a texture stack for each f, and the final texture T^f is computed texel by texel. For each texel stack, the final color is defined as a weighted sum of the input colors. The weight $w^{f_{uv}}$ associated with texel $[u\ v]$ is a combination of sub-weights modeling classical building texture issues: modeled occlusions ($w_{\text{MO}}^{f_{uv}}$), nonmodeled occlusions ($w_{\text{NMO}}^{f_{uv}}$), and spatial resolution ($w_{\text{SR}}^{f_{uv}}$). We now explain in the subsections below the formulations of these sub-weights.

5.2.1 Modeled occlusions

Modeled occlusions refer to the visibility maps that can be constructed for each texture of the stack, using only the camera pose and the buildings 3D model. This information is computed during the texture stack construction step. The weight $w_{\text{MO}}^{f_{uv}}$ associated with each texel equals 1 if it is visible and 0 otherwise. If the pixel with coordinates $[u\ v\ 1]^\top \sim \mathbf{H}_k^{f^{-1}}[u'\ v'\ 1]^\top$ is outside image I_k boundaries, then the texel is not visible. Moreover, if the pixel belongs to the image, the façade f' to which the pixel belongs is determined by back-projection using the camera pose. If $f \neq f'$, it is not visible either (see Fig. 8).

$$w_{\text{MO}}^{f_{uv}} = \begin{cases} 1, & \text{if the texel is visible} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

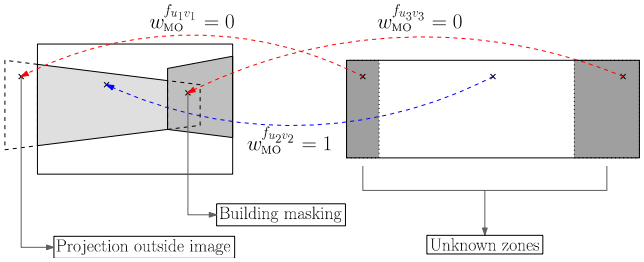


Fig. 8 Detecting nonvisible texels using the 3D model projection

5.2.2 Non modeled occlusions

Let $\mathsf{T}^{f_{uv}}$ be the texel stack with coordinates $[u\ v]^\top$ in T^f. Some of these texels may be erroneous (in the color sense) because of occluding objects nonmodeled in the GIS database. They are called *outliers*. We want to discard these outliers so that only the texels corresponding to the façade (the *inliers*) are used to compute the final texel color in T.

Under the assumption that for a given texel stack $\mathsf{T}^{f_{uv}}$ the outliers represent less than 50% of the samples, then the texel $\mathsf{T}_j^{f_{uv}}$ whose color is the median color of the stack is an inlier. Texels in the stack whose color is "sufficiently close" to the median value are also considered as inliers. This closeness measure is now explained.

Let $C(\mathsf{T}_j^{f_{uv}})$ be the color of the jth texel of the stack. The median color is given by:

$$C\left(\mathsf{T}^{f_{uv}}\right)_{\text{med}} = \underset{j}{\text{med}}\left(C\left(\mathsf{T}_j^{f_{uv}}\right)\right) \tag{5}$$

Deviation of the inliers to the true color is then robustly computed by measuring the median absolute deviation of colors from $C(\mathsf{T}^{f_{uv}})_{\text{med}}$:

$$\begin{aligned}\Delta C\left(\mathsf{T}^{f_{uv}}\right) &= \text{MAD}\left(C\left(\mathsf{T}^{f_{uv}}\right)\right)\\ &= \underset{j}{\text{med}}\left(\left|C\left(\mathsf{T}_j^{f_{uv}}\right) - C\left(\mathsf{T}^{f_{uv}}\right)_{\text{med}}\right|\right)\end{aligned} \tag{6}$$

The texels that are tagged as inliers (noted $\overline{\mathsf{T}}^{f_{uv}}$) are those whose color deviation to $C(\mathsf{T}^{f_{uv}})_{\text{med}}$ is below $\lambda\Delta C(\mathsf{T}^{f_{uv}})$, λ being a scalar fixed to 2 in our case.

$$\begin{aligned}&\forall k, \mathsf{T}_k^{f_{uv}} \in \overline{\mathsf{T}}^{f_{uv}}\\ &\Leftrightarrow \left|C\left(\mathsf{T}_k^{f_{uv}}\right) - C\left(\mathsf{T}^{f_{uv}}\right)_{\text{med}}\right| \leq \lambda\Delta C\left(\mathsf{T}^{f_{uv}}\right)\end{aligned} \tag{7}$$

Outliers removal is simply performed by setting a zero weight $w_{\text{NMO}}^{f_{uv}}$ to these texels:

$$w_{\text{NMO}}^{f_{uv}} = \begin{cases} 1, & \text{if } \mathsf{T}_k^{f_{uv}} \in \overline{\mathsf{T}}^{f_{uv}} \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

5.2.3 Spatial resolution

The final color of texel T_{uv} is now computed using the inliers stack $\overline{\mathsf{T}}^{f_{uv}}$. Each inlier is assigned a weight $w_{\text{SR}}^{f_{uv}}$ so that texels with higher spatial resolution have a greater influence in the final color computation than those with lower resolution. Several criteria can be used to measure this resolution. One option is to estimate the resolution quality using the 3D configuration of the scene. Another one is to measure it directly in the input images.

Distance and angle In [18, 20], view angles are used. More precisely, the angle $\boldsymbol{\theta}$ between the considered pixel line of view and the façade normal is used. The wider this angle is, the lighter the weight $w_{\text{SR}_{an}}^{f_{uv}}$ will be. This is illustrated in Fig. 9: The point with maximal resolution in I_k is $\mathbf{x}_\perp$, corresponding to the 3D point $\mathbf{X}_\perp$ which is the projection of the camera center $\mathbf{C}$. For a given 3D point $\mathbf{X}$ on the façade, the resolution of its 2D point $\mathbf{x}$ decreases as the angle $\widehat{\mathbf{XCX}_\perp}$ increases, due to perspective projection. Assuming that $\theta \in]-\frac{\pi}{2}; \frac{\pi}{2}[$, we pose:

$$w_{\text{SR}_{an}}^{f_{uv}} = \cos|\theta| \tag{9}$$

We believe that in general the angle measure alone is not sufficient to characterize the texels resolution. The distance between the façade and the camera is also of great importance. We thus combine $w_{\text{SR}_{an}}^{f_{uv}}$ to a distance-driven weight $w_{\text{SR}_{di}}^{f_{uv}}$, defined as the (metric) distance between the camera center $\mathbf{C}$ and the 3D point $\mathbf{X}$ corresponding to the considered pixel.

$$w_{\text{SR}_{di}}^{f_{uv}} = \frac{1}{\|\mathbf{C} - \mathbf{X}\|_2} \tag{10}$$

One way to define the spatial resolution weight is then to multiply $w_{\text{SR}_{an}}^{f_{uv}}$ and $w_{\text{SR}_{di}}^{f_{uv}}$.

Projection area Another way to measure the texels resolution is to compute, for each of them, the area in I_k of the corresponding quadrilateral (see Fig. 10). The visual quality of the reconstructed texel is directly proportional to this area. If $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, and $\mathbf{d}$ are the coordinates of the texel "corners" in texture T^f, then the corresponding coordinates $\mathbf{a}'$, $\mathbf{b}'$, $\mathbf{c}'$, and $\mathbf{d}'$ in I_k are given by the homography $\mathbf{H}_k^{f^{-1}}$. The weight $w_{\text{SR}_{ar}}^{f_{uv}}$ is computed as the area of the quadrilateral $(\mathbf{a}', \mathbf{b}', \mathbf{c}', \mathbf{d}')$,

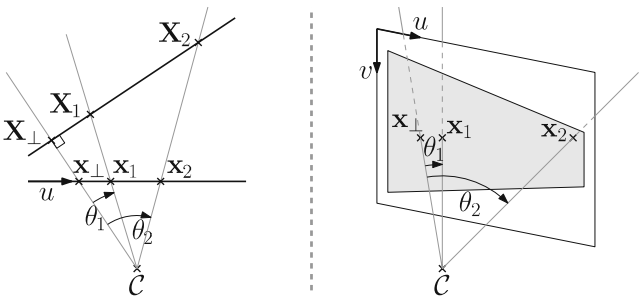


Fig. 9 Top view (*left*) and perspective view (*right*) of a façade projected on an image. Here the 2D point $\mathbf{x_2}$ has lower resolution than point $\mathbf{x_1}$ since angle θ_1 is inferior to angle θ_2

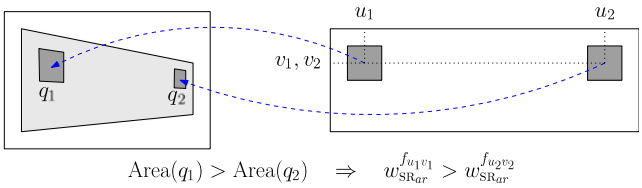


Fig. 10 Area measure of the texels projected in their corresponding image

i.e., as the half norm of the cross product of its diagonals:

$$w_{\text{SR}_{ar}}^{f_{uv}} = \frac{1}{2} \left\| (\mathbf{a}' - \mathbf{c}') \times (\mathbf{b}' - \mathbf{d}') \right\|_2 \tag{11}$$

5.2.4 Fusion procedure

Due to the binary nature of weights $w_{\text{MO}}^{f_{uv}}$ and $w_{\text{NMO}}^{f_{uv}}$, subweights corresponding to a texel stack are not multiplied at once. Instead, the fusion procedure consists in successive steps of outliers removal followed by a weighted sum of the estimated inliers. First, the weights $w_{\text{MO}}^{f_{uv}}$ are used to discard nonvisible texels from the stack. Then, nonmodeled outliers are removed from the remaining texels, leading in the inliers list $\overline{\mathsf{T}}^{f_{uv}}$. With both resolution weighting methods (area or distance plus angle), the weights $w_{\text{SR}}^{f_{uv}}$ are normalized so that their sum equals to one, providing the final weights formulation:

$$w^{f_{uv}} = \begin{cases} w_{\text{SR}_{an}}^{f_{uv}} \cdot w_{\text{SR}_{di}}^{f_{uv}} \Big/ \sum_{i=0}^{M} w_{\text{SR}_{an}i}^{f_{uv}} \cdot w_{\text{SR}_{di}i}^{f_{uv}} & \text{with dist/angle} \\ w_{\text{SR}_{ar}}^{f_{uv}} \Big/ \sum_{i=0}^{M} w_{\text{SR}_{ar}i}^{f_{uv}} & \text{with area} \end{cases} \tag{12}$$

Notice that $M = \text{card}(w_{\text{SR}\ i}^{f_{uv}}) = \text{card}(\overline{\mathsf{T}}^{f_{uv}})$ is the number of inliers texels. The desired texel value is finally computed as:

$$\mathsf{T}_{uv} = \sum_{i=0}^{M} w_{\text{SR}_{ar}i}^{f_{uv}} * \mathsf{T}_{uvi} \tag{13}$$

5.3 Results

First of all, this section presents texture fusion results for a synthetic video (for which the image-model registration is known). We remind briefly the successive steps of the fusion process and illustrate the stack fusion after each step:

1. Textures extraction from the video ..... Fig. 11
2. Modeled occlusions removal ..... Fig. 12b
3. Nonmodeled occlusions removal ..... Fig. 12c
4. Weighting of the valid texels of the texture stack:
   - Either using the distance-angle measure ..... Fig. 12d
   - Or using the projection area measure .. Fig. 12e

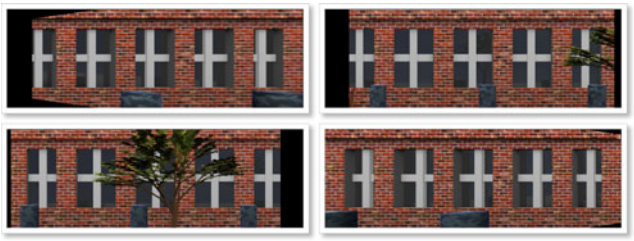
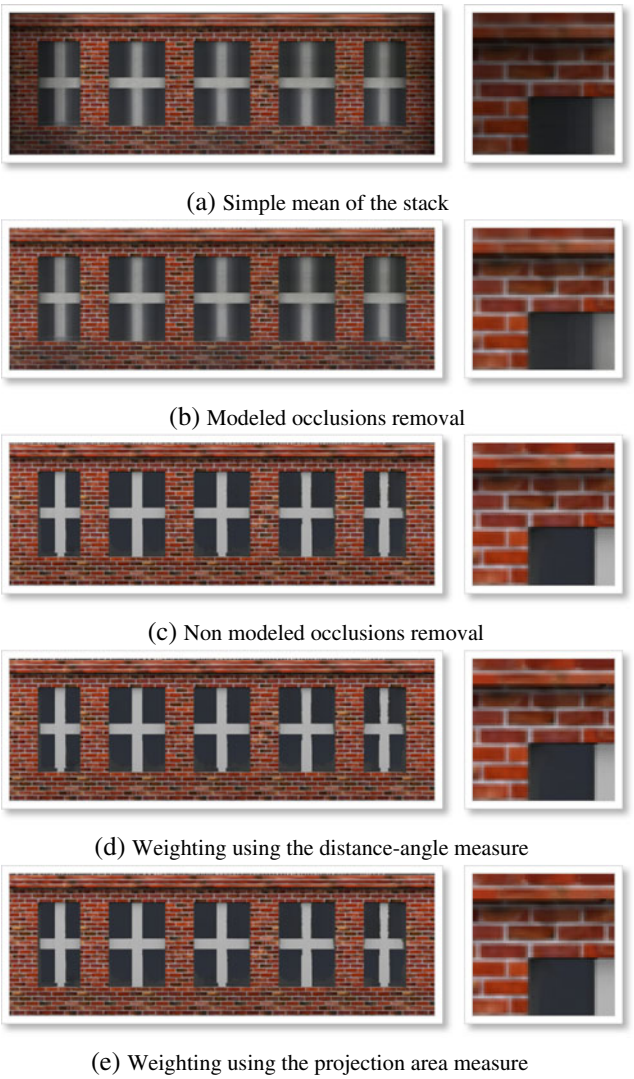


Fig. 11 Extracted textures example



(a) Simple mean of the stack

(b) Modeled occlusions removal

(c) Non modeled occlusions removal

(d) Weighting using the distance-angle measure

(e) Weighting using the projection area measure

Fig. 12 Textures fusion - intermediate steps and final results

Fusion results are also presented for real images in Fig. 13. In this case, the 2D/3D registration used for textures extraction is computed automatically and thus is less accurate than in the synthetic case. Here the projection area measure is used for the weighing step. It leads to results similar to the distance-angle measure since most of the texture accuracy is already obtained after the nonmodeled occlusions removal phase. Projection area is also a less memory consuming process. In Figs. 12 and 13, the left column shows complete textures while the second one highlights local details.

In Fig. 12, for the synthetic video, one can notice that most of the outlier texels are removed during the nonmodeled occlusions step (which is the desired result). The final step dealing with spatial resolution has a quite small influence, and the two resolution measures lead to very similar results. For instance, we observe a *peak signal-to-noise ratio* (PSNR) of 35.421 dB between the image of Fig. 12c and the final texture computed using the distance-angle measure. It is equal to 35.215 dB with the image computed through the projection area measure. The PSNR between the two final results (with the different measures) is equal to 45.340 dB, meaning the human eye may not notice the differences between the two textures. Both spatial resolution measures providing very similar results, the projection area can be used as a reference resolution measure since it can be computed faster than the distance and angle measure.

We also notice that the texture is correctly estimated for the zones belonging to the façade principal plane (those matching the plane in the 3D model). On the contrary, they are blurred either because of the parallax effect (Fig. 12e, top part) or because of noise close to contours (Fig. 12e, windows), for the micro-structure

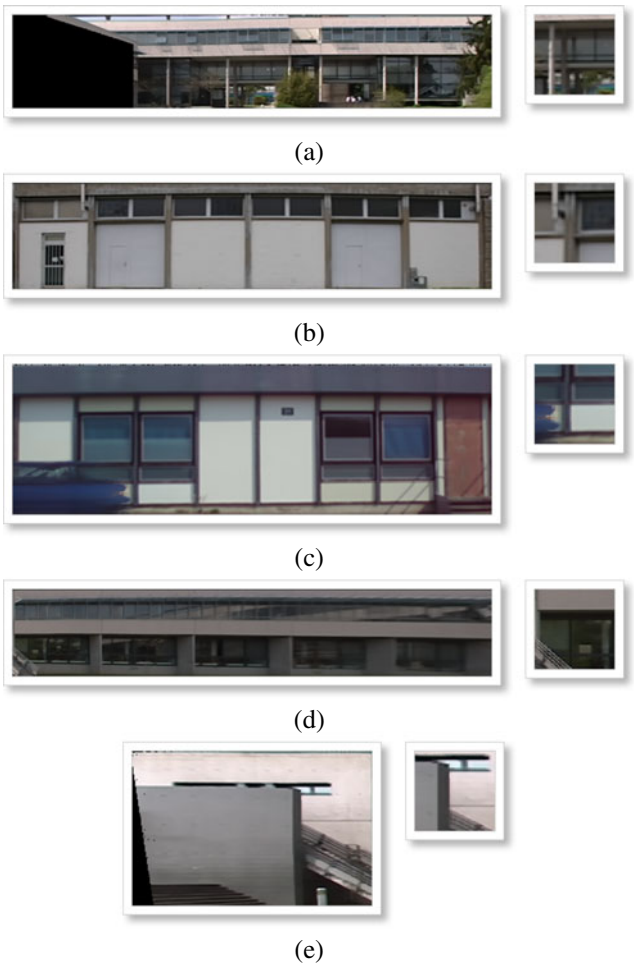


Fig. 13 Textures fusion—RHTF. The *left column* illustrated real textures computed using the projection area measure. The *right* one represents a small detail of each of these textures. We can notice that occluding objects are correctly removed (when they represent less than 50% of a given texel stack) and that contours are preserved, which are both desired results

Fig. 14 Texture fusion—algorithm by Teller et al.. The *left column* presents the textures computed using Teller's algorithm. The *central column* represents a small detail from each of these textures, and the *right column* the same detail computed with our algorithm (best viewed in the electronic version of the paper)

Fig. 15 Comparison between the real view (*left*) and the same view with the superimposed 3D buildings

elements nondescribed in the 3D model. However, this noisy contours problem does not appear in the real sequences, where the structure details are preserved (see for instance the building's poles in Fig. 13a or the stair guardrail in Fig. 13e). The main difference between the synthetic and real videos comes from the fact that for the synthetic videos nonmodeled occlusions have a more important influence.

We can also point out that occluding objects are correctly removed. For the real sequences, this result is particularly visible in Fig. 13c, where the concrete plots on the ground disappear, so does the metallic rail on the right part of the façade. However, the car stays partly visible. This is caused by the fact that its left part hides the façade in every single image of the video.

Finally, we can notice that the specularities are removed in the final textures (Figs. 12 and 13c, d).

In Fig. 14, we compare the results of our RHTF with the method presented by Teller et al. [18, 20]. This method has already been successfully applied to generate high-quality textures for instance the Tech Square at the MIT. RHTF has several advantages over Teller's method:

- It better preserve contours (Fig. 14a vs. Fig. 12e; Fig. 14e vs. Fig. 13e)
- Occluding objects are removed more efficiently using the presented MAD outliers removal method (Fig. 14a vs. Fig. 12e; Fig. 14c vs. Fig. 13c)

This is mainly due to the fact that in Teller's method, iterative correlation masks are used to remove nonmodeled occlusions which are outperformed by the simple median-based measures used in RHTF and to the fact

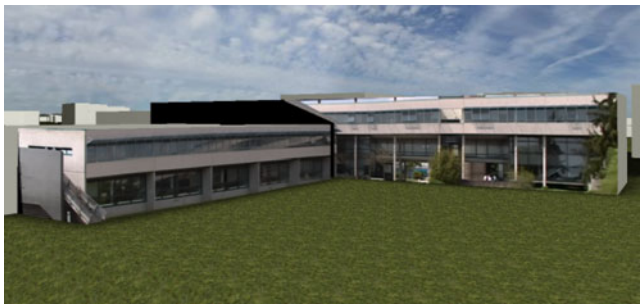


Fig. 16 Virtual view synthesis of the textured GIS database

that while dealing with spatial resolution, Teller's algorithm does not take into account façade distance to the camera but only the viewing angle. Another point is that small errors are more visible with the presented testing images since they are captured "closer" to the buildings than Teller's test images are.

Finally, Fig. 15 illustrates the difference between original images and the corresponding image with the superimposed textured and registered 3D models, together with a fully synthetic view (Fig. 16) whose textures are built upon a single video.

6 Conclusion

This paper presents a new framework allowing registration of GIS-based 3D models with videos, in an automatic way, exploiting GPS measures of the camera position. The key point of initial registration is also considered. Contrary to most existing approaches, we propose an automatic solution with supervised validation for initial pose computation. We also show how such a registration could be exploited to derive automatically photo-realistic façades textures from ground images. The proposed method summarizes the different state-of-the algorithms, allowing the computation of façade textures free from outlier objects, with clear contours and from which specularities are removed. This is a mandatory step to build high-quality and realistic virtual tours within urban environments.

In a more constrained framework, for instance if accurate positional measures are available (e.g., with inertial sensors associated to GPS measures), the supervision phase could be skipped to achieve completely automatic GIS/video registration.

We could also increase registration robustness over time. To do so, the line-based pose computation process may be integrated during the tracking phase. Moreover, structure from motion results from the initial pose

estimation step could also be exploited so as to match not only 2D/3D primitives but also 3D/3D ones (points in this case), for instance in an *iterative closest point* framework. However, since pose computation itself with visual servoing is robust enough—even in ill-posed cases (pure rotation, forward or backward motion with the epipole in images)—such SfM approach would eventually be reliable to improve the pose *initialization* before refinement through visual servoing, not for the accurate pose estimation at each time instant.

In our RHTF, we could try to unify the distance–angle and area distances in a single measure based on pixel-wise solid angles, as the former two produce similar results. The computation of the solid angles can be performed using Girard's theorem and requires the estimation of the dihedral angles formed by the consecutive projected pixel sides. The gain of using such measure is not trivial however, since the area distance remains simpler and faster to evaluate than dihedral angles.

References

1. Colleu T, Sourimant G, Morin L (2008) Automatic initialization for the registration of GIS and video data. In: 3DTV 2008. Istanbul, Turkey
2. Comport A (2005) Robust real-time 3D tracking of rigid and articulated objects for augmented reality and robotics. PhD thesis, Université de Rennes 1, Mention informatique
3. David P, DeMenthon D, Duraiswami R, Samet H (2002) Softposit: simultaneous pose and correspondence determination. In: ECCV (3), pp 698–714
4. David P, Dementhon D, Duraiswami R, Samet H (2004) Softposit: simultaneous pose and correspondence determination. Int J Comput Vis 59(3):259–284
5. Debevec PE, Taylor CJ, Malik J (1996) Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. Comput Graph 30:11–20
6. Fischler MA, Bolles RC (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun ACM 24(6):381–395
7. Gomez JFV, Simon G, Berger MO (2005) Calibration errors in augmented reality: a practical study. In: Proceedings of the fourth IEEE and ACM international symposium on mixed and augmented reality (ISMAR)
8. Karner K, Bauer J, Klaus A, Schindler K (2002) Metropogis: a city information system. In: ICIP02, pp III:533–536
9. Korah T, Rasmussen C (2006) Improving spatiotemporal inpainting with layer appearance models. In: International symposium on visual computing
10. Liu L, Stamos I (2005) Automatic 3D to 2D registration for the photorealistic rendering of urban scenes. In: CVPR '05, pp 137–143
11. Mayer H, Bornik A, Bauer J, Karner KF, Leberl F (2001) Multiresolution texture for photorealistic rendering. In: Spring conference on computer graphics
12. Ortin D, Remondino F (2005) Occlusion-free image generation for realistic texture mapping. In: 3D-ARCH 2005: virtual reconstruction and visualization of complex architectures
13. Poulin P, Ouimet M, Frasson MC (1998) Interactively modeling with photogrammetry. In: Proceedings of eurographics workshop on rendering, vol 98, pp 93–104
14. Rau J, Teo T, Chen L, Tsai F, Hsiao K, Hsu W (2006) Integration of gps, GIS and photogrammetry for texture mapping in photo-realistic city modeling. In: Pacific-Rim symposium on image and video technology, pp 1283–1292
15. Reitmayr G, Drummond T (2006) Going out: robust model-based tracking for outdoor augmented reality. In: ISMAR, pp 109–118
16. Shum HY, Han M, Szeliski R (1998) Interactive construction of 3D models from panoramic mosaics. In: Proc. of CVPR'98
17. Sourimant G, Morin L, Bouatouch K (2007) GPS, GIS and video registration for building reconstruction. In: ICIP 2007, 14th IEEE international conference on image processing. San Antonio, USA
18. Taillandier F (2000) Texture and relief estimation from multiple georeferenced images. Master's thesis, Ecole Polytechnique
19. Teller S, Antone M, Bodnar Z, Bosse M, Coorg S, Jethwa M, Master N (2003) Calibrated, registered images of an extended urban area. Int J Comput Vis 53(1):93–107
20. Wang X, Totaro S, Taillandier F, Hanson A, Teller S (2002) Recovering facade texture and microstructure from real-world images. In: European conference on computer vision